Nagarjuna College of Engineering & Technology
(An Autonomous College under VTU)
Department of Information Science & Engineering

Python Programming (20CSI44)

Laboratory Manual

Sl.No	Title of the experiment
1	Implement a Python Program to find GCD of two numbers.
2	Implement a Python Program to find the square root of a number by Newton's Method.
3	Implement a Python program to find the exponentiation of a number.
4	Implement a Python Program to find the maximum from a list of numbers.
5	Implement a Python Program to perform Linear Search.
6	Implement a Python Program to perform Binary Search.
7	Implement a Python Program to perform Selection sort.
8	Implement a Python Program to perform Insertion sort.
9	Implement a Python Program to perform Merge sort.
10	Implement a Python Program to find first n Prime numbers.

Course Teacher :

Prof. Suresha S,
Assistant Professor,
Dept. of ISE,
NCET.

1) Implement a Python Program to find GCD of two numbers.

```
def gcd(a, b):
    if(b == 0):
        return a
    else:
        return gcd(b, a % b)

a = int(input("enter a value :"))
b = int(input("enter b value :"))

print("GCD of a and b is:", end="")
print(gcd(a, b))
```

Output:

```
enter a value :12
enter b value :14
GCD of a and b is:2
>>>
========== RESTART: C:/
enter a value :15
enter b value :12
GCD of a and b is:3
>>>
========== RESTART: C:/
enter a value :12
enter b value :16
GCD of a and b is:4
```

2) Implement a Python Program to find the square root of a number by Newton's Method.

Newton's Method:

0.5*(approx+n/approx) is the Newton method to find the square root of the number.

```
def newton_method(number, number_iters = 100):
    a = float(number)
    for i in range(number_iters):
        number = 0.5 * (number + a / number)
    return number
```

```
a=int(input("Enter first number:"))
b=int(input("Enter second number:"))
print("Square root of first number:",newton_method(a))
print("Square root of second number:",newton_method(b))
```

Output:

Enter first number:16

Enter second number:4

Square root of first number: 4.0

Square root of second number: 2.0

3) Implement a Python program to find the exponentiation of a number.

```
Base=3
Exponent=4
Print(“Exponential Value is:”, pow(Base,Exponent))
```

Output:

Exponential Value is: 8

OR

```
import math
exponent=4
Print(“Exponential Value is:”, math.exp(exponent))
```

Output:

Exponential Value is: 54.5981500331

OR

```
num=int(input("Enter number: "))

exp=int(input("Enter exponential value: "))

result=1

for i in range(1,exp+1):

    result=result*num

print("Result is:",result)
```

Output:

```
Enter number: 2
Enter exponential value: 10
Result is: 1024
```

OR

```
num=int(input("Enter number: "))

exp=int(input("Enter exponential value: "))

result=num**exp

print("Result is:",result)
```

Output:

```
Enter number: 8
Enter exponential value: 3
Result is: 512
```

OR

```
import math

num=int(input("Enter number: "))

exp=int(input("Enter exponential value: "))

result=math.pow(num,exp)

print("Result is:",result)
```

Output:

```
Enter number: 8
Enter exponential value: 3
Result is: 512
```

4) Implement a Python Program to find the maximum from a list of numbers.

```
# creating empty list
list1 = []                  # asking number of elements to put in list
num = int(input("Enter number of elements in list: "))  # iterating till num to append elements in list
for i in range(1, num + 1):
    ele = int(input("Enter elements: "))
    list1.append(ele)

print("Largest element is:", max(list1))       # print maximum element
```

Output:

Enter number of elements in list: 4

Enter elements: 12

Enter elements: 19

Enter elements: 1

Enter elements: 99

Largest element is: 99

OR
Without using built in functions in python:

```
def myMax(list1):
        max = list1[0]

        for x in list1:
                if x > max :
                max = x
        return max
list1 = [10, 20, 4, 45, 99]
print("Largest element is:", myMax(list1))
```

Output:

Largest element is: 99

5) Implement a Python Program to perform Linear Search.

```
# Linear Search in Python
def linearSearch(array, n, x):                # Going through array sequencially
    for i in range(0, n):
        if (array[i] == x):
            return i
    return -1


array = [2, 4, 0, 1, 9]
x=int(input("enter element to be found: "))
n = len(array)
result = linearSearch(array, n, x)
if(result == -1):
    print("Element not found")
else:
    print("Element found at index: ", result)
```

Output:

```
enter element to be found: 9
Element found at index:  4


enter element to be found: 4
Element found at index:  1

enter element to be found: 8
Element not found
```

6) Implement a Python Program to perform Binary Search.

```
def binarySearch(array, x, low, high):

        while low <= high:                  # Repeat until the pointers low and high meet each other
                mid = low + (high - low)//2
                if array[mid] == x:
                        return mid
                elif array[mid] < x:
                        low = mid + 1
                else:
                        high = mid - 1
                        return -1

array = [3, 4, 5, 6, 7, 8, 9]
x = int(input("enter the value to be searched: "))
result = binarySearch(array, x, 0, len(array)-1)

if result != -1:
   print("Element is present at index ", str(result))
else:
   print("Not found")
```

Output:

```
enter the value to be searched: 8
Element is present at index :  5

enter the value to be searched: 9
Element is present at index :  6

enter the value to be searched: 10
Not found
```

8) Implement a Python Program to perform Insertion sort.

```
def insertion_sort(alist):
    for i in range(1, len(alist)):
        temp = alist[i]
        j = i - 1
        while (j >= 0 and temp < alist[j]):
            alist[j + 1] = alist[j]
            j = j - 1
        alist[j + 1] = temp

alist = input('Enter the list of numbers: ').split()
alist = [int(x) for x in alist]
insertion_sort(alist)
print('Sorted list: ', end='')
print(alist)
```

Output:

```
Enter the list of numbers: 5 4 2 3 1
Sorted list: [1, 2, 3, 4, 5]

Enter the list of numbers: 5
Sorted list: [5]

Enter the list of numbers: 66 7
Sorted list: [7, 66]
```

9) Implement a Python Program to perform Merge sort.

```
def merge(arr, l, m, r):
    n1 = m - l + 1
    n2 = r - m

    L = [0] * (n1)                    # create temp arrays
    R = [0] * (n2)

    for i in range(0, n1):            # Copy data to temp arrays L[] and R[]
        L[i] = arr[l + i]

    for j in range(0, n2):
        R[j] = arr[m + 1 + j]

    i = 0     # Initial index of first subarray          # Merge the temp arrays back into arr[l..r]
    j = 0     # Initial index of second subarray
    k = l     # Initial index of merged subarray
```

```
    while i < n1 and j < n2:
        if L[i] <= R[j]:
            arr[k] = L[i]
            i += 1
        else:
            arr[k] = R[j]
            j += 1
        k += 1

    while i < n1:                 # Copy the remaining elements of L[], if there are any
        arr[k] = L[i]
        i += 1
        k += 1

    while j < n2:                 # Copy the remaining elements of R[], if there are any
        arr[k] = R[j]
        j += 1
        k += 1

# l is for left index and r is right index of the
# sub-array of arr to be sorted
def mergeSort(arr, l, r):
    if l < r:
        m = l+(r-l)//2                    # Same as (l+r)//2, but avoids overflow for  large l and h

 mergeSort(arr, l, m)                          # Sort first and second halves
 mergeSort(arr, m+1, r)
 merge(arr, l, m, r)
 arr = [12, 11, 13, 5, 6, 7]              # Driver code to test above
 n = len(arr)
 print("Given array is")
 for i in range(n):
     print("%d" % arr[i],end=" ")

 mergeSort(arr, 0, n-1)
 print("\n\nSorted array is")
 for i in range(n):
     print("%d" % arr[i],end=" ")
```

Output :

```
Given array is
12 11 13 5 6 7

Sorted array is
5 6 7 11 12 13
```

10) Implement a Python Program to find first n Prime numbers.

```
ower_value = int(input ("Please, Enter the Lowest Range Value: "))
upper_value = int(input ("Please, Enter the Upper Range Value: "))

print ("The Prime Numbers in the range are: ")
for number in range (lower_value, upper_value + 1):
   if number > 1:
      for i in range (2, number):
         if (number % i) == 0:
            break
      else:
         print (number)
```

Output:

```
Please, Enter the Lowest Range Value: 2
Please, Enter the Upper Range Value: 100
The Prime Numbers in the range are:
2
3
5
7
11
13
17
19
23
29
31
37
41
43
47
53
59
61
67
71
73
79
83
89
97
```